CMPSC265 Data Structures and Algorithms
Homework 5

Due: Sunday, Oct 6th, 2019, 11:59pm

Learning Objectives

- Applications on Singly linked list
- Applications on Doubly linked list

Deliverables:

- The modified java file *RemoveLinkedList.java* for Problem 1
- The modified java file *LinkedListCycle.java* for Problem 2
- The modified java file *GroupLinkedList.java* for Problem 3
- A Java File *Dequeue.java* for Problem 4

Instructions:

- Please first download the Java source file for the first three problems, modify the codes based upon requirements, and submit the updated source file.
- Please add the required *credit comments* and *comments* to each Java program you submit.

Problem Specifications:

Problem 1: RemoveLinkedList class (25')

Description:

Given a singly linked list, remove the n-th node from the end of the list.

Example:
Given a linked list: 1->2->3->4->5, and n=2
After removing the second node from the end, the linked list would become:
1->2->3->5

Outputs:

Your output should look something like follows.

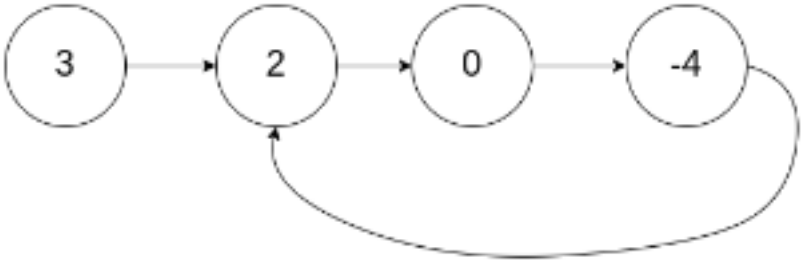
```
$ java RemoveLinkedList
List (first-->last): {5} {4} {3} {2} {1}
Please enter a number N, so that the Nth node from the end
will be deleted:  3
List (first-->last): {5} {4} {2} {1}
```

Problem 2: ***LinkedListCycle class (20')***

Description:

Given a singly linked list, write a program to determine whether it has a cycle.

For example, the following linked list has a cycle.



Hint: you can introduce two pointers traversing the linked list from head with different speeds (like two people running on a loop with different speed). If there is a cycle, the two pointers will finally meet with each other (the person runs faster will catch up the one who runs slower). Otherwise, there would be no cycle.

Outputs:

Your outputs should look something as follows.

```
$ java LinkedListCycle
Does the linked list contain a cyle?: true
```

Problem 3: GroupLinkedList class (25')

Description:

Given a singly linked list, group all odd nodes (not the value of node, but the node number) together followed by all the even nodes.

Example:
Input: 1->2->3->4->5->null
Output: 1->3->5->2->4->null

You should do it in-place, namely, you will NOT create another linked lists or arrays, and then add the nodes back to form the new linked list.

Outputs:

Your outputs should look something as follows.

```
$ java GroupLinkedList
List (first-->last): {1} {2} {3} {4} {5}
List (first-->last): {1} {3} {5} {2} {4}
```

Problem 4: Deque class (30')

Description

Deque is an abstract data type defined by the following structure and operations. A deque is structured as a collection of data items where data items can be added and removed from either end, either front or rear.

Please write a Java class for Deque of Integers, **using doubly linked list**, so that the following method can be implemented:

Methods	**Description**
Deque()	Constructor. A new deque will be instantiated that is empty.
addFront(int)	Add a new integer to the front of the deque; Returns nothing
addRear(int)	Add a new integer to the rear of the deque; Returns nothing
removeFront()	Remove and return the integer at the front from the deque
removeRear()	Remove and return the integer at the end from the deque.
isEmpty()	Tests to see whether the deque is empty Returns a boolean
display()	Show all the integers currently in the deque from front to rear
size()	Returns the number of integers in the deque Returns an integer.

Outputs:

Your implement should support the following operations, and obtain corresponding results.

Operation	**Deque content**	**Return type**
d.isEmpty()	[]	True
d.addRear(4)	[4]	
d.addRear(6)	[4, 6]	
d.addFront(0)	[0, 4, 6]	
d.addFront(10)	[10, 0, 4, 6]	
d.size()	[10, 0, 4, 6]	4
d.isEmpty()	[10, 0, 4, 6]	False
d.removeRear()	[10, 0, 4]	6
d.removeFront()	[0, 4]	10
d.isEmpty()	[0, 4]	False